빌드시 사용되는 환경변수 등의 주요내용 상세 기제

BackEnd

1.SpringBoot

1. 버전
   - SpringSecurity 사용(pom.xml에 사용이 됨)

2. 백엔드 빌드

appication.properties작성 유의사항

- mysql 연결 mysql을 사용을 하여 작업을 진행을 함
- `jwt.header=Authorization` , `jwt.secret=PRIVATE_KEY`
- jwt.header=Authorization , jwt.secrte=PRIVATE_KEY작성 해야함
- `famillink.flask.domain=localhost` 로 작성 해야함
- JWT TOKEN만료시간 관련 작성 사항

```
jwt.token-validity-in-minutes=1
jwt.refresh-validity-in-minutes=10000
```

- 이메일 인증시 관련 작성 사항 : gmail을 이용한 smtp프로토콜 사용
- SpringBoot 파일업로드 / 다운로드 관련 변수

```
spring.servlet.multipart.location=upfiles
spring.servlet.multipart.maxFileSize=50MB
spring.servlet.multipart.maxRequestSize=50MB
```

3.파일 업로드, 다운로드시 유의사항

- 영상파일은 mp4
- 사진 파일은 jpg
- 얼굴 학습 모델은 h5
- 얼굴 학습 label은 txt

확장자로 보내고 저장이 되어야 한다

2.Docker

aws상에 Docker를 설치를 하여 그위에 Springboot, react, mysql, flask의 container를 두어 작업을 진행을 함

Docker 설치 방법

- **Set up the repository**

```
sudo apt-get update
sudo apt-get install \
    ca-certificates \
    curl \
    gnupg \
    lsb-release
```

- Add Docker’s official GPG key:

```
sudo mkdir -m 0755 -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
```

- Use the following command to set up the repository:

```
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

Install Docker Engine

- Update the `apt` package index:

  ```
  sudo apt-get update
  ```

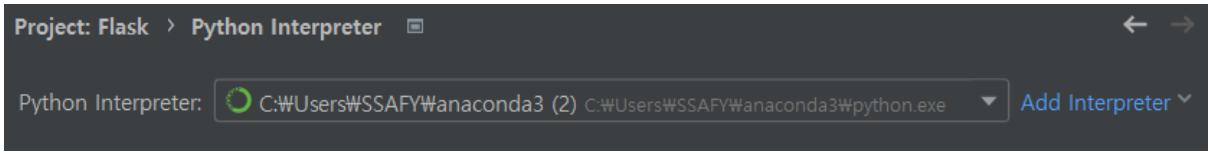
- Install Docker Engine, containerd, and Docker Compose.

  ```
  sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
  ```

3.FLASK

- 파이썬 사용을 위한 pycharm설치
  - https://www.jetbrains.com/ko-kr/pycharm/
- Anaconda
  - https://docs.anaconda.com/anaconda/install/windows/

- pycharm 에서 추가적인 설치 항목들



- python interpreter에 anaconda를 설치를 해준다
- 그후 필요한 것들을 설치를 해준다.
  - `import numpy as np`
  - `from keras.models import load_model`
  - `from flask import Flask, request`
  - `import json`
  - `import os`
  - `from werkzeug.utils import secure_filename`

  ⇒와 같은 것들을 설치

4.젠킨스

- 젠킨스 관련 플러그인
  - **Email Extension버전2.94**
  - **GitLab버전1.7.6**
  - **Jackson 2 API버전**
  - **JAXB 2.3.8-1**
  - JUnit버전1177.v90374a_ef4d09
  - Mailer버전448.v5b_97805e3767
  - **OkHttp버전4.10.0-117.vf786fdfa_22c0**

- 환경 설정
  - 오래된 빌드 삭제 ⇒ Stragegy : Log Rotation ⇒ 보관할 최대 갯수 : 10
  - 소스 코드 관리 ⇒Repository URL: https://lab.ssafy.com/s08-webmobile2-sub2/S08P12A208.git ⇒Credentials: 깃lab 아이디 비밀번호 입력
  - Branches to build ⇒ */master
  - 빌드 유발
    - Build when a change is pushed to GitLab. GitLab webhook URL: http://i8a208.p.ssafy.io:8080/project/docker_compose
    - Push Events
    - Opened Merge Request Events
    - Approved Merge Requests (EE-only)
    - Comments
    - Poll SCM
    - 위 해당 부분들을 체크

- Build Steps
  - **Execute shell**
    - Command

```
docker-compose down --rmi all
cp /home/env/application.properties ./BE/Spring/src/main/resources
docker-compose up -d

docker image prune -a -f
```

AWS 관련 세팅

Nginx Default 값

- port는 80
- document root는 index.html
- 초기 페이지는 index.html 페이지 입니다.

EC2 세팅

- Docker

    - Docker 설치
    - Docker 설치후 Jenkins 컨테이너 설치
    - 그 후 Docker Compose 를 통하여 git lab master branch에 올라가게 될시 clone해와서 react,mysql,spring 컨테이너 생성
    - Docker Network 로 해서 다른 컨테이너 연결

- Mysql
    - port:3306
    - volumes : `/mysql:/var/lib/mysql`